**安装nginx**



wget http://nginx.org/download/nginx-1.12.2.tar.gz

tar xf nginx-1.12.2.tar.gz

cd nginx-1.12.2

groupadd www

useradd  -g www www -M -s /sbin/nologin

yum install openssl-devel pcre-devel gcc gcc-c++ -y

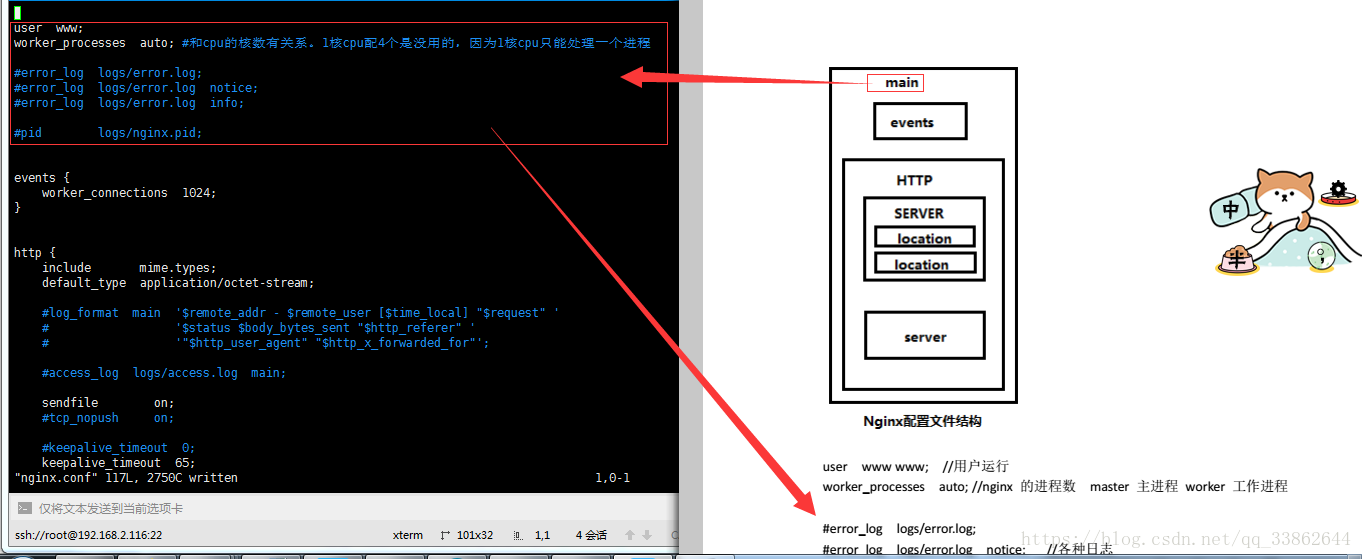
./configure --user=www --group=www --prefix=/usr/local/nginx --with-http\_stub\_status\_module --with-http\_ssl\_module  --with-http\_gzip\_static\_module  --with-http\_sub\_module --with-pcre

make && make install

### 配置文件大致解释

总结：配置文件是嵌套式的，http块中配置server（虚拟主机）

其他的配置方式，如伪静态、gzip压缩等



user  www www;  //用户运行

worker\_processes  auto; //nginx 的进程数  master 主进程 worker 工作进程

#error\_log  logs/error.log;

#error\_log  logs/error.log  notice;   //各种日志

#error\_log  logs/error.log  info;

pid   /var/run/nginx.pid;       //进程pid存放位置

# 修改 文件描述符 ulimit -SHn 65535  /etc/profile

events {

    worker\_connections  65535;     //每个work进程打开的最大连接数

}

http {

公共的配置项,除server外,全是公共的配置项

    server {

    }

}

http{

//文件扩展名与文件类型映射表

include       mime.types;  //location对应的，后缀和值都在mime.types

default\_type  application/octet-stream; #默认文件类型

//设置日志类型 访问日志

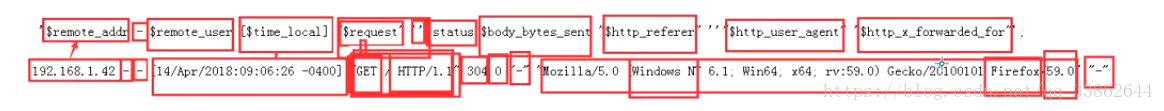
#log\_format  main  '$remote\_addr - $remote\_user [$time\_local] "$request" '

  #                  '$status $body\_bytes\_sent "$http\_referer" '

  #                  '"$http\_user\_agent" "$http\_x\_forwarded\_for"';

  #access\_log  logs/access.log  main;

 //下图是，变量和访问日志里值的对比



sendfile  on  //普通应用为on  下载io的为off

    #客户端请求单个文件的最大字节数

    client\_max\_body\_size 8m; //上传文件 默认是1M

    #防止网络阻塞

    tcp\_nopush on;

    tcp\_nodelay on;

#客户端相关设置

    keepalive\_timeout 60; //连接超时时间，单位是秒

    client\_header\_timeout 10;  #客户端请求头读取超时时间

    client\_body\_timeout 10; #设置客户端请求主体读取超时时间

    send\_timeout 10; #响应客户端超时时间

#gzip模块设置

    gzip on; #开启gzip压缩输出

    gzip\_min\_length 1k; #最小压缩文件大小

    gzip\_buffers 4 16k; #压缩缓冲区

    gzip\_http\_version 1.0; #压缩版本（默认1.1，前端如果是squid2.5请使用1.0）

    #压缩等级 1-9 等级越高，压缩效果越好，节约宽带，但CPU消耗大

    gzip\_comp\_level 2;

    #压缩类型，默认就已经包含text/html，所以下面就不用再写了，写上去也不会有问题，但是会有一个warn。

    gzip\_types text/plain application/x-javascript text/css application/xml;

    #前端缓存服务器缓存经过压缩的页面

#虚拟主机定义

    server {

        listen       80; #监听端口

        server\_name  localhost; #访问域名

        #charset koi8-r; #编码格式，若网页格式与此不同，将被自动转码

        #access\_log  logs/host.access.log  main; #虚拟主机访问日志定义

        location / { #对URL进行匹配

            root   html; #访问路径，可相对也可绝对路径

            index  index.html index.htm;  #首页文件。以下按顺序匹配

        }

        #error\_page  404              /404.html; #错误信息返回页面

        # redirect server error pages to the static page /50x.html

        error\_page   500 502 503 504  /50x.html;

        location = /50x.html {

            root   html;

        }

#访问URL以.php结尾则自动转交给127.0.0.1

        # proxy the PHP scripts to Apache listening on 127.0.0.1:80

        #location ~ \.php$ {

        #    proxy\_pass   http://127.0.0.1;

        #}

#php脚本请求全部转发给FastCGI处理

        # pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000

        #location ~ \.php$ {

        #    root           html;

        #    fastcgi\_pass   127.0.0.1:9000;

        #    fastcgi\_index  index.php;

        #    fastcgi\_param  SCRIPT\_FILENAME  /scripts$fastcgi\_script\_name;

        #    include        fastcgi\_params;

        #}

#禁止访问.ht页面 （需ngx\_http\_access\_module模块）

        # deny access to .htaccess files, if Apache's document root

        # concurs with nginx's one

        #location ~ /\.ht {

        #    deny  all;   //禁止后缀为.ht的访问，和apache那权限差不多

        #}

    }

}

## 基于端口/域名配置多站点：

**基于端口的多站点配置（域名不变，端口变）**

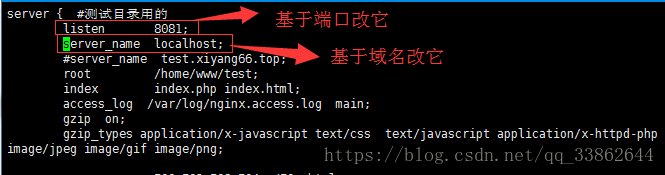
使用场景：1、内网测试；2、接口（不同功能模快，弄个站点，调用）

注意：访问时，拿ip带上不同的端口号

**基于域名的多站点配置（端口不变，域名变）端口也可以写成不一样的。**

1、一台服务器放多个项目，都可以在公网进行访问

注意：访问时，使用不同的域名（因为端口都是80）



### 反向代理

